# Abstract Cofibered Domains: Application to the Alias Analysis of Untyped Programs

Arnaud Venet

LIX, École Polytechnique, 91128 Palaiseau, France.
venet@lix.polytechnique.fr

**Abstract.** We present a class of domains for Abstract Interpretation, the cofibered domains, that are obtained by "glueing" a category of partially ordered sets together. The internal structure of these domains is well suited to the compositional design of approximations and widening operators, and we give generic methods for performing such constructions. We illustrate the interest of these domains by developing an alias analysis of untyped programs handling structured data. The results obtained with this analysis are comparable in accuracy to those obtained with the most powerful alias analyses existing for typed languages.

## 1 Introduction

Widening operators have originally been used in Abstract Interpretation [CC77] in order to cope with infinite domains on which abstract iteration sequences were not necessarily computable (e.g. [CC76, CH78]). In fact, the notion of widening is much more powerful since it allows the definition of abstract interpretations with very few hypotheses on their structure by *dynamically* constructing the abstract domain [Cou78, CC92a, Cou96]. In this paper we introduce a new class of such abstract domains: the *cofibered domains.*

A cofibered domain consists of a category of partially ordered sets "glued" together in a sense that we will make clear. Like for other composite domains in Abstract Interpretation (e.g. the reduced product of lattices [CC79]), most constructions over a cofibered domain can be achieved *compositionally* by combining the corresponding constructions over its base components. We illustrate the interest of this approach by constructing an alias analysis for a small untyped language with structured data. This analysis gives results comparable in accuracy to those obtained with Deutsch's framework [Deu92a, Deu92b, Deu94] whose applicability is restricted to languages with explicit datatype declarations.

The paper is organized as follows. In Sect. 2 we recall briefly the context of semantic approximation in which our work takes place. Section 3 is devoted to describing the framework of cofibered domains. We use dynamic partitioning [Bou92] as the running example of our presentation. In Sect. 4 we describe a generic construction of widening operators over cofibered domains. We apply the previous techniques to build an alias analysis for an untyped language in Sect. 5.

## 2 Abstract Interpretation with Widening

Let $P$ be a program of a language $\mathcal{L}$. We suppose that the semantics $\mathcal{S}_P^\natural$ of $P$ is given by the least fixpoint of a $\sqcup^\natural$-complete endomorphism $F^\natural$ over a complete lattice $(\mathcal{D}^\natural, \sqsubseteq^\natural, \perp^\natural, \sqcup^\natural, \top^\natural, \sqcap^\natural)$, where $\mathcal{D}^\natural$ is the *concrete semantic domain* which expresses program properties or behaviours. This is a quite common situation in practice but it is possible to relax significantly the previous hypotheses [CC92a].

Abstract Interpretation provides general frameworks to reason about semantic approximation [CC92a]. We choose one of these that is quite general and well-suited for our purpose, but all the techniques developed in this paper can be adapted to other frameworks. More precisely, following [CC92a] we assume that an abstract semantic specification of $P$ is given by a preordered set $(\mathcal{D}, \preceq)$, the *abstract semantic domain*, related to $\mathcal{D}^\natural$ by a *concretization function* $\gamma : \mathcal{D} \longrightarrow \mathcal{D}^\natural$, an *abstract basis* $\perp \in \mathcal{D}$, and an *abstract semantic function* $F : \mathcal{D} \longrightarrow \mathcal{D}$ such that:

(i) $\perp^\natural \sqsubseteq^\natural \gamma(\perp)$.
(ii) $\forall x, y \in \mathcal{D} : x \preceq y \Longrightarrow \gamma(x) \sqsubseteq^\natural \gamma(y)$.
(iii) $\forall x \in \mathcal{D} : F^\natural \circ \gamma(x) \sqsubseteq^\natural \gamma \circ F(x)$.

In order to compute an approximation $\mathcal{S}_P$ of the concrete semantics of $P$ in $\mathcal{D}$ we introduce the notion of *widening operator.*

**Definition 1 Widening operator [CC77, CC92a].** A *widening* on $\mathcal{D}$ is a binary operator $\nabla : \mathcal{D} \times \mathcal{D} \longrightarrow \mathcal{D}$ which satisfies the following properties:

W1- $\forall x, y \in \mathcal{D} : x \preceq x \nabla y$.
W2- $\forall x, y \in \mathcal{D} : y \preceq x \nabla y$.
W3- For every sequence $(x_n)_{n>0}$ of elements of $\mathcal{D}$, the sequence $(x_n^\nabla)_{n>0}$ inductively defined as follows:

$$\begin{cases} x_0^\nabla &= x_0 \\ x_{n+1}^\nabla &= x_n^\nabla \nabla x_{n+1} \end{cases}$$

is ultimately stationary.

□

*Example 1* **Intervals [CC76].** Let $(\mathcal{D}_\mathrm{I}, \subseteq)$ be the domain of intervals where $\mathcal{D}_\mathrm{I} = \{\emptyset\} \cup \{[a,b] \mid a \in \mathbb{Z} \cup \{-\infty\} \wedge b \in \mathbb{Z} \cup \{+\infty\}\}$ and $\subseteq$ is set inclusion. The widening $\nabla_\mathrm{I}$ is defined as follows:

$$\begin{cases} I \nabla_\mathrm{I} \emptyset = I \\ \emptyset \nabla_\mathrm{I} I = I \\ [a_1, b_1] \nabla_\mathrm{I} [a_2, b_2] = [\text{if } a_2 < a_1 \text{ then } -\infty \text{ else } a_1, \\ \qquad\qquad\qquad \text{if } b_1 < b_2 \text{ then } +\infty \text{ else } b_1] \end{cases}$$

In other words, the upper (resp. lower) limit of an interval is extrapolated to $+\infty$ (resp. $-\infty$) whenever it increases (resp. decreases). □

**Proposition 2 Abstract iterates [CC92a, CC92b].** *The abstract iteration sequence* $(F_n^\nabla)_{n>0}$ *given by:*

$$\begin{cases} F_0^\nabla &= \bot \\ F_{n+1}^\nabla &= F_n^\nabla \qquad\qquad \textit{if } F(F_n^\nabla) \preceq F_n^\nabla \\ &= F_n^\nabla \nabla F(F_n^\nabla) \ \textit{otherwise} \end{cases}$$

*is ultimately stationary and its limit* $\mathcal{S}_P$ *satisfies* $\mathcal{S}_P^\natural \sqsubseteq^\natural \gamma(\mathcal{S}_P)$. *Moreover, if* $N \geq 0$ *is such that* $F_N^\nabla = F_{N+1}^\nabla$, *then* $\forall n \geq N : F_n^\nabla = F_N^\nabla$.

## 3 Cofibered Domains

The intuitive idea of "glueing" a category of posets is formalized by the *Grothendieck construction.*

**Definition 3 Grothendieck construction [BW90].** Let $\mathbb{D}$ be a small category and **Pos** be the category of posets and monotone maps. We associate to any functor $\Delta : \mathbb{D} \longrightarrow$ **Pos** the category $\mathbf{G}\Delta$ defined as follows:

(i) An object of $\mathbf{G}\Delta$ is a pair $(D, x)$ where $D$ is an object of $\mathbb{D}$ and $x$ is an element of the poset $\Delta D$.

(ii) An arrow $(D, x) \xrightarrow{f} (E, y)$ of $\mathbf{G}\Delta$ is a morphism $D \xrightarrow{f} E$ of $\mathbb{D}$ such that $\Delta f(x) \sqsubseteq_E y$, where $\sqsubseteq_E$ denotes the order relation on $\Delta E$.

(iii) The composition of two arrows $(D, x) \xrightarrow{f} (E, y)$ and $(E, y) \xrightarrow{g} (F, z)$ is given by $(D, x) \xrightarrow{g \circ f} (F, z)$. It is explicited in the following diagram:

$$\begin{tikzcd}
D \arrow[r, "f"] & E \arrow[r, "g"] & F \\
x \arrow[r, "\Delta f"] & \Delta f(x) \arrow[r, "\Delta g"] \arrow[d, "\sqsubseteq_E"'] & \Delta g \circ \Delta f(x) \arrow[d, "\sqsubseteq_F"'] \\
 & y \arrow[r, "\Delta g"] & \Delta g(y) \arrow[d, "\sqsubseteq_F"'] \\
 & & z
\end{tikzcd}$$

□

**Definition 4 Cofibered domain.** A *cofibered domain*[1] is a preordered set $(\mathcal{D}, \preceq)$ for which there exists a functor $\Delta : \mathbb{D} \longrightarrow$ **Pos** such that:

(i) $\mathcal{D}$ is the set of objects of $\mathbf{G}\Delta$.

(ii) For all $(D, x), (E, y) \in \mathcal{D}$, $(D, x) \preceq (E, y)$ whenever there exists an arrow $(D, x) \xrightarrow{f} (E, y)$ in $\mathbf{G}\Delta$. □

[1] The term *cofibered domain* comes from the fact that such a domain can be endowed with the structure of a *cofibration* [BW90].

In other words, $(\mathcal{D}, \preceq)$ is the preorder obtained by *flattening* the category $\mathbf{G}\Delta$. We call the functor $\Delta$ the *display* associated to the cofibered domain. For each object $D$ in $\mathbb{D}$ we call the poset $\Delta D$ the *fiber of $\Delta$ over $D$*.

*Example 2* **Dynamic partitioning I**. We illustrate cofibered domains by constructing an approximation of the domain $(\wp(C \longrightarrow L), \subseteq)$, where $C \longrightarrow L$ is the set of functions from a set $C$ into a complete lattice $(L, \sqsubseteq, \bot, \sqcup, \top, \sqcap)$. An interesting instance of this problem arises when $C$ is the set of control points of a program (lexical points, stacks, etc.) and $L$ is the powerset of all memory states. This has been extensively studied in [Bou92] where the powerful abstraction framework of *dynamic partitioning* has been developed. In its simplest form it amounts to abstracting a function $\rho : C \longrightarrow L$ by a partial function $\mu$ from $\wp(C)$ into $L$. The value of $\rho$ at a point $x$ is approximated by the join of the values $\mu(X)$ on the subsets $X$ of $C$ that contain $x$.

Let $\mathbf{P}(C)$ be the category whose objects are the subsets of $\wp(C)$. An arrow $D \xrightarrow{f} E$ of $\mathbf{P}(C)$ is a function $f : D \longrightarrow E$ such that for any $X \in D$ we have $X \subseteq f(X)$. Composition and identities are the usual ones. One can easily check that this definition is consistent. We define the functor $\Delta_C : \mathbf{P}(C) \longrightarrow \mathbf{Pos}$ as follows:

- For any object $D$ of $\mathbf{P}(C)$, $\Delta_C D$ is the set $D \longrightarrow L$ with the pointwise ordering.
- If $D \xrightarrow{f} E$ is an arrow in $\mathbf{P}(C)$, $\Delta_C f$ maps any $\mu : D \longrightarrow L$ to the function $\lambda X \cdot \sqcup\{\mu(Y) \mid Y \in f^{-1}(X)\}$ of $\Delta_C E$.

We then construct the cofibered domain $(\mathcal{D}_C, \preceq_C)$ associated to the display $\Delta_C$. An element $(D, \mu)$ of $\mathcal{D}_C$ represents a partial function from $\wp(C)$ into $L$, where $D \subseteq \wp(C)$ is the domain of definition of the function. The approximation of the concrete domain is given by the concretization function $\gamma_C : (\mathcal{D}_C, \preceq_C) \longrightarrow (\wp(C \longrightarrow L), \subseteq)$ that maps any $(D, \mu)$ in $\mathcal{D}_C$ to the set $\{\rho : C \longrightarrow L \mid \forall x \in C : \rho(x) \sqsubseteq \sqcup\{\mu(X) \mid X \in D \wedge x \in X\}\}$. □

In Sect. 2 we defined the connection between the concrete and abstract semantic domains as a concretization function $\gamma : (\mathcal{D}, \preceq) \longrightarrow (\mathcal{D}^\natural, \sqsubseteq)$. However it is quite rare in practice to directly build the abstract domain $(\mathcal{D}, \preceq)$. In general one proceeds by *stepwise* refinements of the approximation: if $(\mathcal{D}^\sharp, \preceq^\sharp)$ is another preordered set and $\gamma^\sharp : (\mathcal{D}^\sharp, \preceq^\sharp) \longrightarrow (\mathcal{D}, \preceq)$ is a monotone map, then the composite $\gamma \circ \gamma^\sharp : (\mathcal{D}^\sharp, \preceq^\sharp) \longrightarrow (\mathcal{D}^\natural, \sqsubseteq)$ is a further approximation of $(\mathcal{D}^\natural, \sqsubseteq)$. If $(\mathcal{D}, \preceq)$ is cofibered via a display $\Delta$ one can obtain an approximation of $(\mathcal{D}, \preceq)$ by means of a *fiberwise approximation* of $\Delta$. In order to describe precisely this idea we need to introduce the notion of lax natural transformation.

**Definition 5 Lax natural transformation [Kel74].** A *lax natural transformation* (lax n.t. for short) $\Delta^\sharp \overset{\kappa}{\rightsquigarrow} \Delta$ between two functors $\Delta^\sharp, \Delta : \mathbb{D} \longrightarrow \mathbf{Pos}$ is given by a morphism $\Delta^\sharp D \xrightarrow{\kappa D} \Delta D$ for each object $D$ in $\mathbb{D}$ and by a collection of commutative diagrams:

$$\begin{tikzcd}
\Delta^\sharp D \arrow[r, "\Delta^\sharp f"] \arrow[d, "\kappa_D"'] \arrow[dr, phantom, "\sqsubseteq_E"] & \Delta^\sharp E \arrow[d, "\kappa_E"] \\
\Delta D \arrow[r, "\Delta f"'] & \Delta E
\end{tikzcd}$$

□

The intuition is that a morphism $\kappa_D$ is a local concretization function that relates the fiber $\Delta D$ with its approximation $\Delta^\sharp D$. The commutative diagram above means that $\Delta f \circ \kappa_D \sqsubseteq_E \kappa_E \circ \Delta^\sharp F$, i.e. $\Delta^\sharp f$ is a sound approximation of $\Delta f$.

Let $(\mathcal{D}, \preceq)$ and $(\mathcal{D}^\sharp, \preceq^\sharp)$ be two cofibered domains and $\Delta : \mathbb{D} \longrightarrow \mathbf{Pos}$, $\Delta^\sharp : \mathbb{D}^\sharp \longrightarrow \mathbf{Pos}$ be the associated displays. A *fiberwise approximation* of $(\mathcal{D}, \preceq)$ by $(\mathcal{D}^\sharp, \preceq^\sharp)$ is given by a functor $\Gamma : \mathbb{D}^\sharp \longrightarrow \mathbb{D}$ and a lax n.t. $\Delta^\sharp \overset{\kappa}{\rightsquigarrow} \Delta \circ \Gamma$ expressed diagrammatically by:

$$\begin{tikzcd}
\mathbb{D} \arrow[dr, "\Delta"', ""{name=L}] & & \mathbb{D}^\sharp \arrow[ll, "\Gamma"'] \arrow[dl, "\Delta^\sharp", ""'{name=R}] \\
& \mathbf{Pos} &
\arrow[from=R, to=L, squiggly, "\kappa"']
\end{tikzcd}$$

$\Gamma$ can be seen as a "concretization functor" and $\mathbb{D}^\sharp$ as an abstraction of the shape of the cofibered domain $(\mathcal{D}, \preceq)$. The functor $\Gamma$ is the *global* part of the approximation and the lax n.t. $\kappa$ is the *local* part. We can then "glue" these two parts in order to obtain a concretization function $\mathbf{G}(\Gamma, \kappa) : (\mathcal{D}^\sharp, \preceq^\sharp) \longrightarrow (\mathcal{D}, \preceq)$.

**Proposition 6.** *The function* $\mathbf{G}(\Gamma, \kappa) : \mathcal{D}^\sharp \longrightarrow \mathcal{D}$ *that sends any* $(D^\sharp, x^\sharp)$ *in* $\mathcal{D}^\sharp$ *to* $(\Gamma D^\sharp, \kappa_{D^\sharp}(x^\sharp))$ *is monotone.*

*Example 3* **Dynamic partitioning II**. We carry on with Example 2. We suppose that we are provided with two approximations $\gamma_C : (C^\sharp, \subseteq^\sharp) \longrightarrow (\wp(C), \subseteq)$ and $\gamma_L : (L^\sharp, \sqsubseteq^\sharp) \longrightarrow (L, \sqsubseteq)$. We suppose that $L^\sharp$ has the structure of a $\sqcup^\sharp$-semilattice $(L^\sharp, \sqsubseteq^\sharp, \perp^\sharp, \sqcup^\sharp)$ and that $\gamma_C$ is injective (this is always possible, see [CC79]). Now let $\mathbf{P}_f(C^\sharp)$ be the category whose objects are finite subsets of $C^\sharp$. An arrow $D^\sharp \xrightarrow{f} E^\sharp$ of $\mathbf{P}_f(C^\sharp)$ is a function $f : D^\sharp \longrightarrow E^\sharp$ such that for any $X^\sharp \in D^\sharp$ we have $X^\sharp \subseteq^\sharp f(X^\sharp)$. Let $\Delta^\sharp_C : \mathbf{P}_f(C^\sharp) \longrightarrow \mathbf{Pos}$ be the functor defined as follows:

- For any $D^\sharp$ in $\mathbf{P}_f(C^\sharp)$, $\Delta^\sharp_C D^\sharp$ is the set $D^\sharp \longrightarrow L^\sharp$ with the pointwise ordering.

– If $D^\sharp \xrightarrow{f} E^\sharp$ is an arrow in $\mathbf{P}_f(C^\sharp)$, $\Delta^\sharp_C f$ maps any $\mu^\sharp : D^\sharp \longrightarrow L^\sharp$ to the function $\lambda X^\sharp \cdot \sqcup^\sharp\{\mu^\sharp(Y^\sharp) \mid Y^\sharp \in f^{-1}(X^\sharp)\}$ of $\Delta^\sharp_C E^\sharp$.

Let $\Gamma : \mathbf{P}_f(C^\sharp) \longrightarrow \mathbf{P}(C)$ be the functor that sends any object $D^\sharp$ to $\{\gamma_C(X^\sharp) \mid X^\sharp \in D^\sharp\}$ and any arrow $D^\sharp \xrightarrow{f} E^\sharp$ to the function that maps $\gamma_C(X^\sharp)$ to $\gamma_C(f(X^\sharp))$ for every $X^\sharp \in D^\sharp$. This definition is not ambiguous because $\gamma_C$ is injective. We now define a lax n.t. $\Delta^\sharp_C \overset{\kappa}{\rightsquigarrow} \Gamma \circ \Delta_C$. Let $D^\sharp$ be in $\mathbf{P}_f(C^\sharp)$, $\kappa_{D^\sharp}$ sends every function $\mu^\sharp$ in $\Delta^\sharp_C(D^\sharp)$ to the function $\mu$ that maps $\gamma_C(X^\sharp)$ to $\gamma_L \circ \mu^\sharp(X^\sharp)$, for every $X^\sharp$ in $D^\sharp$. We denote by $(\mathcal{D}^\sharp_C, \preceq^\sharp_C)$ the cofibered domain associated to the display $\Delta^\sharp_C$. By proposition 6 we obtain a fiberwise approximation $\gamma : (\mathcal{D}^\sharp_C, \preceq^\sharp_C) \longrightarrow (\mathcal{D}_C, \preceq_C)$ from $(\Gamma, \kappa)$. □

## 4 Construction of Widenings on Cofibered Domains

Let $(\mathcal{D}, \preceq)$ be a cofibered domain with display $\Delta : \mathbb{D} \longrightarrow \mathbf{Pos}$. We suppose that for every object $D$ in $\mathbb{D}$ the fiber $\Delta D$ is provided with a widening operator $\nabla_D : \Delta D \times \Delta D \longrightarrow \Delta D$ satisfying conditions W1, W2 and W3 of Definition 1. Now we need to define a notion of widening on the category $\mathbb{D}$.

**Definition 7 $\omega$-chain.** An *$\omega$-chain* in $\mathbb{D}$ is a sequence of arrows $(D_n \xrightarrow{f_n} D_{n+1})_{n \geq 0}$. We say that the $\omega$-chain is *ultimately pseudo-stationary* if there exists a rank $N \geq 0$ such that for all $n \geq N$, $f_n$ is an isomorphism. □

A *widening operator* $\nabla$ on $\mathbb{D}$ associates to any two objects $D$, $E$ of $\mathbb{D}$ two arrows:

$$\begin{tikzcd} D \arrow[r, "(D \nabla E)_1"] & D \nabla E & E \arrow[l, "(D \nabla E)_2"'] \end{tikzcd}$$

such that for any sequence of objects $(D_n)_{n \geq 0}$, the $\omega$-chain $(D^\nabla_n \xrightarrow{f^\nabla_n} D^\nabla_{n+1})_{n \geq 0}$ defined by:

$$\begin{cases} D^\nabla_0 = D_0 \\ D^\nabla_{n+1} = D^\nabla_n \nabla D_{n+1} \\ f^\nabla_n = (D^\nabla_n \nabla D_{n+1})_1 \end{cases}$$

is ultimately pseudo-stationary. Moreover we require $\nabla$ to be stable under isomorphism, that is, whenever $D \cong D'$ and $E \cong E'$, then $D \nabla E \cong D' \nabla E'$.

**Definition 8 Widening on cofibered domains.** We assume that $\mathbb{D}$ is provided with a widening operator $\nabla$. If $(D, x)$ and $(E, y)$ are elements of $\mathcal{D}$, we define $(D, x) \overline{\nabla} (E, y)$ as follows:

– $(D, x) \ \overline{\nabla} \ (E, y) \overset{\text{def}}{=} (D, x \nabla_D \Delta((D \nabla E)_1^{-1} \circ (D \nabla E)_2)(y))$ if $(D \nabla E)_1$ is an isomorphism. This is expressed by the following diagram:

$$\begin{tikzcd} D \arrow[d, dash] & D \nabla E \arrow[l, "(D \nabla E)_1^{-1}"'] & E \arrow[l, "(D \nabla E)_2"'] \arrow[d, dash] \\ x \nabla_D \Delta((D \nabla E)_1^{-1} \circ (D \nabla E)_2)(y) & & y \arrow[ll, dashed] \end{tikzcd}$$

- $(D,x)\ \overline{\nabla}\ (E,y) \overset{\text{def}}{=} (D \nabla E, \Delta(D\nabla E)_1(x)\ \nabla_{D\nabla E}\ \Delta(D\nabla E)_2(y))$ otherwise. That is, graphically:

$$\begin{tikzcd}
D \arrow[r, "(D \nabla E)_1"] \arrow[d, dash] & D \nabla E \arrow[d, dash] & E \arrow[l, "(D \nabla E)_2"'] \arrow[d, dash] \\
x \arrow[r, dashed] & \Delta(D\nabla E)_1(x)\ \nabla_{D\nabla E}\ \Delta(D\nabla E)_2(y) & y \arrow[l, dashed]
\end{tikzcd}$$

□

Intuitively the first case means that when the fiber is "stable", i.e. $(D\nabla E)_1$ is an isomorphism, we "transfer" the abstract property $y$ into the fiber and we make the widening with $x$. Otherwise we transfer $x$ and $y$ into the fiber over $D \nabla E$ and we make the widening in this fiber.

**Theorem 9.** *$\overline{\nabla}$ is a widening operator (i.e. it satisfies conditions W1, W2 and W3 of definition 1).*

*Example 4* **Dynamic partitioning III**. We apply this method to the cofibered domain of Example 3. We suppose that the semilattice $(L^\sharp, \sqsubseteq^\sharp, \perp^\sharp, \sqcup^\sharp)$ comes equipped with a widening operator $\nabla_{L^\sharp}$. For each $D^\sharp$ in $\mathbf{P}_f(C^\sharp)$ we define a widening $\nabla_{D^\sharp}$ on the fiber over $D^\sharp$ by pointwise application of $\nabla_{L^\sharp}$. It is not possible to exhibit a widening for $\mathbf{P}_f(C^\sharp)$ in whole generality since it strongly depends on the nature of $C^\sharp$. Therefore we treat a particular instance of $C$ and $C^\sharp$. We suppose that we are analyzing an imperative program $P$ with a set $\mathcal{L}$ which labels the instructions in $P$. We denote by $\mathcal{C}$ the subset of $\mathcal{L}$ consisting of all labels of procedure calls. We define a control point to be a pair $(S,\ell)$ where $S \in \mathcal{C}^*$ is a stack of procedure calls and $\ell \in \mathcal{L}$ denotes the current program point. $C$ is the collection of all sets of control points. We do not specify the domain $L$ introduced in Example 2, which is intended to represent sets of memory states.

We define $C^\sharp$ to be the poset $\mathcal{L} \times (\mathcal{C} \longrightarrow \mathcal{D}_{\mathrm{I}})$, where $\mathcal{D}_{\mathrm{I}}$ is the domain of intervals defined in Example 1, the ordering being given componentwise. The injective function $\gamma_C$ sends a pair $(\ell,\nu)$ to the set $\{(S,\ell) \mid \forall c \in \mathcal{C} : |S|_c \in \nu(c)\}$, where $|S|_c$ denotes the number of occurrences of the label $c$ in $S$. That is, we abstract a stack by the number of times each procedure call occurs in it. We define a widening operator $\nabla_C$ on $C^\sharp$ by componentwise application of $\nabla_{\mathrm{I}}$. If $D^\sharp$ is an element of $\mathbf{P}_f(C^\sharp)$ and $\ell \in \mathcal{L}$, we denote by $D^\sharp(\ell)$ the element $\dot{\sqcup}_{\mathrm{I}}\ \{\mu \mid (\ell,\mu) \in D^\sharp\}$ of $C^\sharp$, where $\dot{\sqcup}_{\mathrm{I}}$ is the pointwise extension of the join $\sqcup_{\mathrm{I}}$ on the lattice $\mathcal{D}_{\mathrm{I}}$ of intervals. Now let $D_1^\sharp$ and $D_2^\sharp$ be elements of $\mathbf{P}_f(C^\sharp)$. We define the widening $D_1^\sharp \nabla D_2^\sharp$ as follows:

$$D_1^\sharp \nabla D_2^\sharp \overset{\text{def}}{=} \{(\ell, D_1^\sharp(\ell) \nabla_{\mathrm{I}} D_2^\sharp(\ell)) \mid \ell \in \mathcal{L}\}$$

For $i \in \{1,2\}$, $(D_1^\sharp \nabla D_2^\sharp)_i$ sends any $(\ell,\mu)$ in $D_i^\sharp$ to $(\ell, D_1^\sharp(\ell) \nabla_{\mathrm{I}} D_2^\sharp(\ell))$ in $D_1^\sharp \nabla D_2^\sharp$. We readily check that this defines a widening on the category $\mathbf{P}_f(C^\sharp)$. By applying Theorem 9 we obtain a widening $\overline{\nabla}_C$ on the cofibered domain $\mathcal{D}_C^\sharp$.

Note that the construction of such a widening (and the proof of its validity) would have been rather intricate without using the cofibered structure of the domain. □

## 5 Application to the Alias Analysis of Untyped Programs

We apply the previous techniques to sketch the construction of an alias analysis for a small imperative language without datatype declarations. We consider a program $P$ written in this language. Let $\mathcal{C}$ be the set of data constructors occurring in $P$ and $\Sigma$ be the set of associated data selectors. If $f \in \mathcal{C}$ has arity $n$, we denote by $f_1, \ldots, f_n$ the corresponding data selectors. We will essentially focus on the two assignment instructions in the language that are involved in the production of aliases: $x := y.w$ and $x := f(x_1, \ldots, x_n)$ where $w \in \Sigma^*$, $f \in \mathcal{C}$ and $x, y$ are elements of the set $V_P$ of variables occurring in $P$. The treatment of control structures (conditionals, loops, recursion) is quite standard [CC77, Gra92] and we do not detail it.

Following [Jon81, Deu92b], a semantic configuration is a pair $(L, \equiv)$ where $L$ is a *prefix-closed* subset of $V_P.\Sigma^*$ and $\equiv$ is a *right-regular equivalence relation* over $L$. Right-regularity means that whenever $v \equiv w$ and $v.a, w.a \in L$, then $v.a \equiv w.a$. $L$ describes the set of access paths in the structures pointed by the variables of the program, and $\equiv$ is the *aliasing* relation on these paths. Right-regularity models equality of pointers in this semantic model.

*Example 5.* Consider the following program:

```
x := nil; y := nil;
for i:= 0 to N do begin
  h := a;
  x := cons(h, x);
  y := cons(h, y);
end;
```

We suppose that the data selectors associated to `cons` are $hd$ and $tl$. Then the semantic configuration at the end of the program is given by $(L, \equiv)$ where $L$ is the set of prefixes of $\{x.tl^n.hd, y.tl^n.hd \mid 0 \leq n \leq N\} \cup \{h\}$ and $\equiv$ is such that $x.tl^n.hd \equiv y.tl^n.hd$ and $h \equiv x.hd \equiv y.hd$. □

We can always assume that none of the variables occurring at the right-hand side of an assignment expression appears at the left-hand side by adding intermediate variables. Let $A \overset{\text{def}}{=} V_P \cup \Sigma$. For any prefix-closed subset $L$ of $A^*$ and any binary relation $\rho$ over $L$, we denote by $[\rho]_L$ the least right-regular equivalence relation over $L$ containing $\rho$. If $w \in A^*$ and $L \subseteq A^*$, we denote by $w^{-1}L$ the language $\{u \in A^* \mid w.u \in L\}$. The semantics $\{\!|x := y.w|\!\}$ of the instruction $x := y.w$ maps $(L, \equiv)$ to $(L_\bullet, \equiv_\bullet)$ where:

- $L_\bullet \overset{\text{def}}{=} (L \backslash x.\Sigma^*) \cup x.((y.w)^{-1}L)$.
- $\equiv_\bullet \overset{\text{def}}{=} [(\equiv \cap (L \backslash x.\Sigma^*)^2) \cup \{(x, y.w)\}]_{L_\bullet}$.

The semantics $\{\!|x := f(x_1, \ldots, x_n)|\!\}$ maps $(L, \equiv)$ to $(L_\bullet, \equiv_\bullet)$ where:

- $L_\bullet \overset{\text{def}}{=} (L \backslash x.\Sigma^*) \cup \bigcup_{1 \leq i \leq n} x.f_i.(x_i^{-1} L)$.
- $\equiv_\bullet \overset{\text{def}}{=} [(\equiv \cap (L \backslash x.\Sigma^*)^2) \cup \{(x.f_i, x_i) \mid 1 \leq i \leq n\}]_{L_\bullet}$.

See [Jon81] for more detail on this kind of semantics. The domain $(\mathcal{D}^\natural, \preceq^\natural)$ associated to the collecting semantics of the program is the powerset of all semantic configurations $(L, \equiv)$ ordered by inclusion.

We first approximate $(\mathcal{D}^\natural, \preceq^\natural)$ by a cofibered domain $(\mathcal{D}, \preceq)$. Let $\wp_<(A^*)$ be the set of all prefix-closed languages over $A$ and $\Delta : (\wp_<(A^*), \subseteq) \longrightarrow \mathbf{Pos}$ be the display that sends any $L \in \wp_<(A^*)$ to the powerset of $\{(L', \rho) \mid L' \subseteq L \land \rho \in \wp(L' \times L')\}$ ordered by inclusion. The image of an arrow $L_1 \subseteq L_2$ is the natural inclusion map of $\Delta L_1$ into $\Delta L_2$. The concretization function $\mathrm{g} : \mathcal{D} \longrightarrow \mathcal{D}^\natural$ maps any $(L, X)$ to $\{(L', \equiv) \mid L' \subseteq L \land (L', \equiv) \in X\}$.

We will make a fiberwise approximation of $\Delta$, but we first need some notations. For any function $f : X \longrightarrow Y$ we denote by $\wp f : (\wp(X), \subseteq) \longrightarrow (\wp(Y), \subseteq)$ its powerset extension. That is, for any $A \in \wp(X)$, $\wp f(A) = \{f(x) \mid x \in A\}$. If $f_1 : X_1 \longrightarrow Y_1$ and $f_2 : X_2 \longrightarrow Y_2$ are two functions, we denote by $f_1 \times f_2 : X_1 \times X_2 \longrightarrow Y_1 \times Y_2$ the function that maps $(x_1, x_2)$ to $(f_1(x_1), f_2(x_2))$.

We choose to abstract the domain $L$ of an alias relation by a regular language. It is represented by an automaton $(Q, I, \tau)$ over $A$ which consists of a finite set of states $Q$, a set of initial states $I \subseteq Q$, and a transition relation $\tau \in \wp(Q \times A \times Q)$. Since $L$ is prefix-closed all states of the automaton are terminal. A morphism $\mathcal{A}_1 \xrightarrow{f} \mathcal{A}_2$ between two automata $\mathcal{A}_1 = (Q_1, I_1, \tau_1)$ and $\mathcal{A}_2 = (Q_2, I_2, \tau_2)$ is given by two functions $f_0 : Q_1 \longrightarrow Q_2$ and $f_1 : \tau_1 \longrightarrow \tau_2$, such that $f_0(I_1) \subseteq I_2$ and for all $(q, a, q') \in \tau_1$, $f_1(q, a, q') = (f_0(q), a, f_0(q'))$. Automata over $A$ with morphisms between them form a category $\mathbb{A}$. In order to keep this category small and representable we suppose that all states come from an infinite and recursively enumerable set $\mathcal{Q}$.

A path $\pi$ of an automaton $\mathcal{A} = (Q, I, \tau)$ is a sequence of adjacent transitions $(q_0, a_0, q_1)(q_1, a_1, q_2) \ldots (q_n, a_n, q_{n+1})$ such that $q_0 \in I$. Let $\mathrm{Paths}(\mathcal{A})$ be the set of paths of $\mathcal{A}$. We denote by $i(\pi)$ the initial state of $\pi$, by $t(\pi)$ its terminal state and by $\ell(\pi)$ the word labelling $\pi$. Let $\mathbf{R} : \mathbb{A} \longrightarrow \mathbf{Pos}$ be the functor that sends any automaton $\mathcal{A}$ to the poset $(\wp(\mathrm{Paths}(\mathcal{A}) \times \mathrm{Paths}(\mathcal{A})), \subseteq)$ of binary relations over paths of $\mathcal{A}$. A morphism of automata $\mathcal{A}_1 \xrightarrow{f} \mathcal{A}_2$ induces a function $f^* : \mathrm{Paths}(\mathcal{A}_1) \longrightarrow \mathrm{Paths}(\mathcal{A}_2)$ in the obvious way. We therefore define $\mathbf{R}f$ to be $\wp(f^* \times f^*)$.

Let $\Lambda : \mathbb{A} \longrightarrow (\wp_<(A^*), \subseteq)$ be the functor that sends any automaton to the language that it recognizes and any morphism to the inclusion arrow. For any automaton $\mathcal{A} = (Q, I, \tau)$ in $\mathbb{A}$, let $\kappa^0_{\mathcal{A}} : \mathbf{R}\mathcal{A} \longrightarrow (\Delta \circ \Lambda)\mathcal{A}$ be the morphism in $\mathbf{Pos}$ that sends any relation $\rho$ in $\mathbf{R}\mathcal{A}$ to the set of pairs $(L, \rho_\bullet)$ for which there exists a function $\lambda \in L \longrightarrow \mathrm{Paths}(\mathcal{A})$ such that:

- $\forall w \in L : \ell(\lambda(w)) = w$.
- $\forall w \in L : \forall a \in A : (w.a \in L) \land (\lambda(w) = \pi) \Longrightarrow \exists \sigma \in \tau : \lambda(w.a) = \pi\sigma$.
- $\forall u, v \in L : (u, v) \in \rho_\bullet \Longrightarrow (u = v) \lor ((\lambda(u), \lambda(v)) \in \rho)$.

**Proposition 10.** *$\kappa^0$ defines a lax natural transformation $\mathbf{R} \rightsquigarrow \Delta \circ \Lambda$.*

Therefore $(\Lambda, \kappa^0)$ induces a fiberwise approximation of $\mathcal{D}$. Following the ideas developed in [Deu92a, Deu92b] we now introduce a numerical abstraction of paths in an automaton. It basically amounts to abstracting a path by the number of times it runs through each arrow of the automaton.

Let $\mathcal{A} = (Q, I, \tau)$ be an automaton of $\mathbb{A}$. We denote by $R(\mathcal{A})$ the set $(I \times Q)^2$. Let $\mathbf{R}^\nu : \mathbb{A} \longrightarrow \mathbf{Pos}$ be the functor that sends an automaton $\mathcal{A} = (Q, I, \tau)$ to the set $R(\mathcal{A}) \longrightarrow \wp((\tau \longrightarrow \mathbb{N})^2)$ ordered by pointwise inclusion. If $V$ and $W$ are two finite sets and $f : V \longrightarrow W$ is a function between them, we define the function $f^\nu : (V \longrightarrow \mathbb{N}) \longrightarrow (W \longrightarrow \mathbb{N})$ as follows:

$$\forall \rho \in (V \longrightarrow \mathbb{N}) : f^\nu(\rho) \overset{\text{def}}{=} \lambda y \cdot \sum_{x \in f^{-1}(y)} \rho(x)$$

Let $\mathcal{A}_1 = (Q_1, I_1, \tau_1)$ and $\mathcal{A}_2 = (Q_2, I_2, \tau_2)$ be two automata and $\mathcal{A}_1 \xrightarrow{f} \mathcal{A}_2$ be a morphism between them. We denote by $f_R : R(\mathcal{A}_1) \longrightarrow R(\mathcal{A}_2)$ the function that maps $(i, t, i', t')$ to $(f_0(i), f_0(t), f_0(i'), f_0(t'))$. We define $\mathbf{R}^\nu f$ as follows:

$$\mathbf{R}^\nu f \overset{\text{def}}{=} \lambda \rho \cdot \lambda r \cdot \{(f_1^\nu \times f_1^\nu)(\mu) \mid \exists r' \in f_R^{-1}(r) : \mu \in \rho(r')\}$$

For any path $\pi$ of an automaton $(Q, I, \tau)$, we denote by $\pi^\circ$ its *commutative image*, that is the function $\pi^\circ : \tau \longrightarrow \mathbb{N}$ that associates to each transition $\sigma \in \tau$ the number $|\pi|_\sigma$ of times it occurs in $\pi$.

For any $\mathcal{A} = (Q, I, \tau)$ in $\mathbb{A}$, let $\kappa^1_{\mathcal{A}} : \mathbf{R}^\nu \mathcal{A} \longrightarrow \mathbf{R}\mathcal{A}$ be the morphism in $\mathbf{Pos}$ that sends any $\rho$ in $\mathbf{R}^\nu \mathcal{A}$ to the element $\{(\pi_1, \pi_2) \mid (\pi_1^\circ, \pi_2^\circ) \in \rho(i(\pi_1), t(\pi_1), i(\pi_2), t(\pi_2))\}$ of $\mathbf{R}\mathcal{A}$. In other words we approximate a path in an automaton by the pair of its initial and final states together with its commutative image.

**Proposition 11.** *$\kappa^1$ defines a lax natural transformation $\mathbf{R}^\nu \rightsquigarrow \mathbf{R}$.*

If $\mathrm{Id}_{\mathbb{A}}$ is the identity functor on $\mathbb{A}$, $(\mathrm{Id}_{\mathbb{A}}, \kappa^1)$ provides a fiberwise approximation of the cofibered domain given by the display $\mathbf{R} : \mathbb{A} \longrightarrow \mathbf{Pos}$.

Now it remains to give a computable approximation of the poset $(\wp(V \longrightarrow \mathbb{N}), \subseteq)$ for any finite set $V$, in order to obtain an effective abstract domain. Several abstractions of this kind have been developed like the *arithmetic intervals* [CC76], the *arithmetic congruences* [Gra89], the *linear equalities* [Kar76], the *linear inequalities* [CH78] or the *linear congruence equalities* [Gra91]. We will not stick to any particular one and leave the choice of the numerical abstraction as a parameter of our domain. We therefore give an abstract description of a numerical domain.

**Definition 12 Abstract numerical domain.** An *abstract numerical domain* $\mathcal{V}^\sharp$ associates to each finite set $V$ a lattice $(\mathcal{V}^\sharp V, \sqsubseteq^\sharp_V, \perp^\sharp_V, \sqcup^\sharp_V, \top^\sharp_V, \sqcap^\sharp_V)$ together with a concretization $\gamma_V : (\mathcal{V}^\sharp, \sqsubseteq^\sharp_V) \longrightarrow (\wp(V \longrightarrow \mathbb{N}), \subseteq)$. If $V$ and $W$ are finite sets and $f : V \longrightarrow W$ is a function between them, there is a computable function $\mathcal{V}^\sharp f : \mathcal{V}^\sharp V \longrightarrow \mathcal{V}^\sharp W$, such that $\wp f^\nu \circ \gamma_V \subseteq \gamma_W \circ \mathcal{V}^\sharp f$. If $V \overset{f}{\rightarrowtail} W$ is an injection, there are two computable functions $\overrightarrow{f} : \mathcal{V}^\sharp V \longrightarrow \mathcal{V}^\sharp W$ and $\overleftarrow{f} : \mathcal{V}^\sharp W \longrightarrow \mathcal{V}^\sharp V$ such that:

- $\forall \nu^\sharp \in \mathcal{V}^\sharp W : \forall \nu \in \gamma_W(\nu^\sharp) : \nu \circ f \subseteq \gamma_V(\overleftarrow{f}(\nu^\sharp))$.
- $\forall \mu^\sharp \in \mathcal{V}^\sharp V : \forall \mu \in \gamma_V(\mu^\sharp) : \{\nu \in W \longrightarrow \mathbb{N} \mid \nu \circ f = \mu\} \subseteq \gamma_W(\overrightarrow{f}(\mu^\sharp))$.

□

Moreover, if $S$ is a system of linear equations over the set of variables $V$, there is a computable element $\mathbf{Sol}^\sharp_V(S)$ of $\mathcal{V}^\sharp V$ which upper-approximates the set of solutions of $S$ in $V \longrightarrow \mathbb{N}$. For all previously cited numerical domains, $\mathcal{V}^\sharp$ can be shown to be a functor from the category **Finset** of finite sets and functions to **Pos**. We will omit the subscript $V$ in the previous definitions whenever it will be clear from the context.

*Example 6.* In Karr's domain [Kar76], an element of $\wp(V \longrightarrow \mathbb{N})$ is upper-approximated by an affine subspace of $\mathbb{Q}^V$, where $\mathbb{Q}$ is the field of rational numbers. An affine subspace of $\mathbb{Q}^V$ can be defined by a system of affine equations over the set of variables $V$. Any function $f : V \longrightarrow W$ induces a linear map $f_{\mathbb{Q}} : \mathbb{Q}^V \longrightarrow \mathbb{Q}^W$. Thus $\mathcal{V}^\sharp f$ is the function that sends any affine subspace of $\mathbb{Q}^V$ to its image by $f_{\mathbb{Q}}$. If $f$ is an injection, $\overleftarrow{f}$ is the orthogonal projection of affine subspaces of $\mathbb{Q}^W$ onto $\mathbb{Q}^V$. If $S$ is a system of equations defining an affine subspace $E$ of $\mathbb{Q}^V$, $\overrightarrow{f}(E)$ is the solution in $\mathbb{Q}^W$ of the system obtained from $S$ by replacing every occurrence of a variable $x$ by $f(x)$. Finally, for any system of affine equations $S$, $\mathbf{Sol}^\sharp_V(S)$ is computable by standard methods. □

If $V$ and $W$ are finite sets we denote by $V \oplus W$ their disjoint union. We will tacitly use the natural isomorphism $(V \longrightarrow \mathbb{N}) \times (W \longrightarrow \mathbb{N}) \overset{\sim}{=} (V \oplus W) \longrightarrow \mathbb{N}$ in the sequel. Let $\mathbf{R}^\sharp : \mathbb{A} \longrightarrow \mathbf{Pos}$ be the functor that sends an automaton $\mathcal{A} = (Q, I, \tau)$ to $(\mathcal{V}^\sharp(\tau \oplus \tau), \sqsubseteq^\sharp_{\tau \oplus \tau})$. Let $\mathcal{A}_1 = (Q_1, I_1, \tau_1)$ and $\mathcal{A}_2 = (Q_2, I_2, \tau_2)$ be two automata and $\mathcal{A}_1 \xrightarrow{f} \mathcal{A}_2$ be a morphism between them. We define $\mathbf{R}^\sharp f$ as follows:

$$\mathbf{R}^\sharp f \overset{\text{def}}{=} \lambda \rho^\sharp \cdot \lambda r \cdot \sqcup^\sharp_{\tau_2 \oplus \tau_2} \{\mathcal{V}^\sharp(f_1^\nu \times f_1^\nu)(\rho^\sharp(r')) \mid r' \in f_R^{-1}(r)\}$$

Now to each automaton $\mathcal{A} = (Q, I, \tau)$ we associate the morphism $\kappa^2_{\mathcal{A}} : \mathbf{R}^\sharp \mathcal{A} \longrightarrow \mathbf{R}^\nu \mathcal{A}$ in **Pos** defined as follows:

$$\kappa^2_{\mathcal{A}} \overset{\text{def}}{=} \lambda \rho^\sharp \cdot \lambda r \cdot \gamma_{\tau \oplus \tau}(\rho^\sharp(r))$$

**Proposition 13.** *$\kappa^2$ defines a lax natural transformation $\mathbf{R}^\sharp \rightsquigarrow \mathbf{R}^\nu$.*

$(\mathrm{Id}_{\mathbb{A}}, \kappa^2)$ induces a fiberwise approximation of the cofibered domain associated to $\mathbf{R}^\nu$. If we denote by $\mathcal{D}^\sharp$ the cofibered domain given by the display $\mathbf{R}^\sharp \longrightarrow \mathbf{Pos}$, we obtain a concretization function $\gamma : \mathcal{D}^\sharp \longrightarrow \mathcal{D}$ by composing all previous approximations.

It now remains to define the abstract semantics of the two assignment instructions over $\mathcal{D}^\sharp$. We first need to define the abstract counterpart to the right-regular equivalence closure operator $[-]_L$. Since the closure of a relation by this operator can be seen as a fixpoint computation, we will use the techniques of Sect. 2 to perform an abstract computation locally over a fiber of $\mathcal{D}^\sharp$.

**Definition 14.** Let $\mathcal{A} = (Q, I, \tau)$ be an automaton. We define the binary relation $\xrightarrow{clo}$ over $\mathbf{R}^\sharp\mathcal{A}$ as follows:

(i) If $\rho^\sharp((q_1, q_2), (q_1', q_2')) = \nu^\sharp$, then $\rho^\sharp \xrightarrow{clo} \rho^\sharp_\bullet$ where

$$\rho^\sharp_\bullet(r) = \begin{cases} \nu^\sharp \text{ if } r = ((q_1', q_2'), (q_1, q_2)) \\ \bot^\sharp \text{ otherwise} \end{cases}$$

(ii) Let $T_1 = T_2 = T_3 = \tau$ and $T = T_1 \oplus T_2 \oplus T_3$. Let $T_1 \oplus T_2 \overset{f_1}{\rightarrowtail} T$, $T_2 \oplus T_3 \overset{f_2}{\rightarrowtail} T$ and $T_1 \oplus T_3 \overset{f}{\rightarrowtail} T$ be the canonical inclusion maps in **Finset**. If $\rho^\sharp((q_1, q_2), (q_1', q_2')) = \nu_1^\sharp$ and $\rho^\sharp((q_1', q_2'), (q_1'', q_2'')) = \nu_2^\sharp$, then $\rho^\sharp \xrightarrow{clo} \rho^\sharp_\bullet$ where

$$\rho^\sharp_\bullet(r) = \begin{cases} \overleftarrow{f}(\overrightarrow{f_1}(\nu_1^\sharp) \sqcap^\sharp \overrightarrow{f_2}(\nu_2^\sharp)) & \text{if } r = ((q_1, q_2), (q_1'', q_2'')) \\ \bot^\sharp & \text{otherwise} \end{cases}$$

(iii) If $\sigma_1 = (q_2, a, q_3)$ and $\sigma_2 = (q_2', a, q_3')$ are in $\tau$, then let $T_1 = T_2 = \tau$; $S_1 = \{\sigma_1\}$, $S_2 = \{\sigma_2\}$ and $T = T_1 \oplus T_2 \oplus S_1 \oplus S_2$. Let $T_1 \oplus T_2 \overset{f_1}{\rightarrowtail} T$ be the canonical inclusion map and $T_1 \oplus T_2 \overset{f_2}{\rightarrowtail} T$ be the inclusion map such that $f_2(\sigma_1) \in S_1$ and $f_2(\sigma_2) \in S_2$. Let $S$ be the system of affine equations over $T$ defined as follows:

$$\begin{cases} f_2(\sigma_1) = f_1(\sigma_1) + 1 \\ f_2(\sigma_2) = f_1(\sigma_2) + 1 \end{cases}$$

If $\rho^\sharp((q_1, q_2), (q_1', q_2')) = \nu^\sharp$, then $\rho^\sharp \xrightarrow{clo} \rho^\sharp_\bullet$ where

$$\rho^\sharp_\bullet(r) = \begin{cases} \overleftarrow{f_2}(\overrightarrow{f_1}(\nu^\sharp) \sqcap^\sharp \mathbf{Sol}^\sharp(S)) & \text{if } r = ((q_1, q_3), (q_1', q_3')) \\ \bot^\sharp & \text{otherwise} \end{cases}$$

Let $\mathrm{F}^\sharp_{clo} \in \mathbf{R}^\sharp\mathcal{A} \longrightarrow \mathbf{R}^\sharp\mathcal{A}$ be the map defined as:

$$\mathrm{F}^\sharp_{clo} = \lambda\rho^\sharp \cdot \bigsqcup\nolimits^\sharp \{\rho^\sharp_\bullet \mid \rho^\sharp \xrightarrow{clo} \rho^\sharp_\bullet\}$$

Every previously cited numerical domain comes with a widening operator on each lattice $\mathcal{V}^\sharp V$ for any finite set $V$. Then, for any $\rho^\sharp$ in $\mathbf{R}^\sharp\mathcal{A}$ we define $[\rho^\sharp]^\sharp_\mathcal{A}$ to be the limit of the iteration sequence with widening of Proposition 2 applied to $\mathrm{F}^\sharp_{clo}$, using $\rho^\sharp$ as a basis for the iteration and the widening defined locally on the fiber $\mathbf{R}^\sharp\mathcal{A}$. □

We also need the abstract counterpart to the quotient operation $w^{-1}L$ over languages.

**Definition 15.** Let $(\mathcal{A}, \rho^\sharp)$ be an element of $\mathcal{D}^\sharp$ and $a \in A$. We define $\mathrm{Elim}_a(\mathcal{A}, \rho^\sharp) \overset{\text{def}}{=} (\mathcal{A}^\sharp_\dagger, \rho^\sharp_\dagger)$ as follows:

- If $\mathcal{A} = (Q, I, \tau)$, then $\mathcal{A}_\dagger \overset{\text{def}}{=} (Q_\dagger, I_\dagger, \tau_\dagger)$, where $Q_\dagger = Q$, $I_\dagger = I$ and $\tau_\dagger = \tau \cap (Q \times (A \backslash \{a\}) \times Q)$.
- If $\tau_\dagger \oplus \tau_\dagger \xrightarrow{f} \tau \oplus \tau$ is the natural inclusion map, then $\rho^\sharp_\dagger \overset{\text{def}}{=} \overleftarrow{f}(\rho^\sharp)$. □

We can now define the abstract semantics of the assignment instructions. In the following we suppose that we are provided with two distinguished elements $\Omega$ and $\Omega'$ in $\mathcal{Q}$. Moreover, if $f \in Q \longrightarrow Q'$ is a function between finite subsets of $\mathcal{Q}$, we will denote by $\overline{f}$ the function $f \times \mathrm{Id}_A \times f \in Q \times A \times Q \longrightarrow Q' \times A \times Q'$.

**Definition 16 Abstract semantics of $x := y.w$.** For any $(\mathcal{A}, \rho^\sharp) \in \mathcal{D}^\sharp$, let $((Q_\bullet, I_\bullet, \tau_\bullet), \rho^\sharp_\bullet) \overset{\text{def}}{=} \mathrm{Elim}_x(\mathcal{A}, \rho^\sharp)$ and $P_\bullet \overset{\text{def}}{=} \{\pi \in \mathrm{Paths}(\mathcal{A}_\bullet) \mid \ell(\pi) = y.w\}$. We put

$$Q_\dagger \overset{\text{def}}{=} Q_\bullet \oplus \bigoplus_{\pi \in P_\bullet} Q_\bullet \oplus \bigoplus_{\pi \in P_\bullet} \{\Omega\}$$

Let $Q_\bullet \xrightarrow{e} Q_\dagger$, $Q_\bullet \xrightarrow{e_\pi} Q_\dagger$ and $\{\Omega\} \xrightarrow{o_\pi} Q_\dagger$, $\pi \in P_\bullet$, be the canonical inclusion maps in **Finset**. We put $I_\dagger \overset{\text{def}}{=} e(I_\bullet) \cup \{o_\pi(\Omega) \mid \pi \in P_\bullet\}$ and $\tau_\dagger \overset{\text{def}}{=} \overline{e}(\tau_\bullet) \cup \{(o_\pi(\Omega), x, e_\pi(t(\pi))) \mid \pi \in P_\bullet\}$. We then define the automaton $\mathcal{A}_\dagger \overset{\text{def}}{=} (Q_\dagger, I_\dagger, \tau_\dagger)$. Let $\mathcal{A}_\dagger \xrightarrow{\iota^1} \mathcal{A}_\dagger \coprod \mathcal{A}_\dagger \xleftarrow{\iota^2} \mathcal{A}_\dagger$ be a coproduct diagram in $\mathbb{A}$ and let $\pi \in P_\bullet$. We define the system of affine equations $S_\pi$ over $\tau_\dagger \oplus \tau_\dagger$ as follows:

$$\begin{cases} \sigma = \begin{cases} 1 & \text{if } \sigma = \iota^2_1(o_\pi(\Omega), x, e_\pi(t(\pi))) \\ |\pi|_{\sigma'} & \text{if } \sigma = \iota^1_1 \circ \overline{e}(\sigma') \\ 0 & \text{otherwise} \end{cases} \\ \sigma \in \tau_\dagger \oplus \tau_\dagger \end{cases}$$

We define the element $\rho^\sharp_\pi$ of $\mathbf{R}^\sharp(\mathcal{A}_\dagger)$ as follows:

$$\rho^\sharp_\pi(r) \overset{\text{def}}{=} \begin{cases} \mathbf{Sol}^\sharp(S_\pi) & \text{if } r = ((\iota^1_0 \circ e(i(\pi)), \iota^1_0 \circ e(t(\pi))), (\iota^2_0 \circ o_\pi(\Omega), \iota^2_0 \circ e_\pi(t(\pi)))) \\ \bot^\sharp_{\tau_\dagger \oplus \tau_\dagger} & \text{otherwise} \end{cases}$$

Finally we put

$$\{\!|x := y.w|\!\}^\sharp(\mathcal{A}, \rho^\sharp) \overset{\text{def}}{=} (\mathcal{A}_\dagger, [\mathcal{V}^\sharp(\overline{e} \oplus \overline{e})(\rho^\sharp_\bullet) \sqcup^\sharp \bigsqcup_{\pi \in P_\bullet}{}^\sharp \rho^\sharp_\pi]^\sharp_{\mathcal{A}_\dagger})$$

□

**Definition 17 Abstract semantics of $x := f(x_1, \ldots, x_n)$.** For any $(\mathcal{A}, \rho^\sharp) \in \mathcal{D}^\sharp$, let $((Q_\bullet, I_\bullet, \tau_\bullet), \rho^\sharp_\bullet) \overset{\text{def}}{=} \mathrm{Elim}_x(\mathcal{A}, \rho^\sharp)$ and $P_\bullet \overset{\text{def}}{=} \{(\pi_1, \ldots, \pi_n) \mid \forall i \in \{1, \ldots, n\} : \pi_i \in \mathrm{Paths}(\mathcal{A}) \wedge \ell(\pi_i) = x_i\}$. We put

$$Q_\dagger \overset{\text{def}}{=} Q_\bullet \oplus \bigoplus_{\alpha \in P_\bullet} \left( \bigoplus_{i=1}^{n} Q_\bullet \right) \oplus \bigoplus_{\alpha \in P_\bullet} \{\Omega, \Omega'\}$$

Let $Q_\bullet \overset{e}{\rightarrowtail} Q_\dagger$, $Q_\bullet \overset{e^i_\alpha}{\rightarrowtail} Q_\dagger$ and $\{\Omega, \Omega'\} \overset{o_\alpha}{\rightarrowtail} Q_\dagger$, $\alpha \in P_\bullet$, $1 \leq i \leq n$, be the canonical inclusion maps in **Finset**. We put $I_\dagger \overset{\text{def}}{=} e(I_\bullet) \cup \{o_\alpha(\Omega) \mid \alpha \in P_\bullet\}$ and $\tau_\dagger \overset{\text{def}}{=} \overline{e}(\tau_\bullet) \cup \{(o_\alpha(\Omega), x, o_\alpha(\Omega')) \mid \alpha \in P_\bullet\} \cup \{(o_\alpha(\Omega'), f_i, e^i_\alpha(t(\pi_i))) \mid \alpha = (\pi_1, \ldots, \pi_n) \in P_\bullet\}$. We then define the automaton $\mathcal{A}_\dagger \overset{\text{def}}{=} (Q_\dagger, I_\dagger, \tau_\dagger)$. Let $\mathcal{A}_\dagger \xrightarrow{\iota^1} \mathcal{A}_\dagger \coprod \mathcal{A}_\dagger \xleftarrow{\iota^2} \mathcal{A}_\dagger$ be a coproduct diagram in $\mathbb{A}$. Let $\alpha = (\pi_1, \ldots, \pi_n) \in P_\bullet$ and $i \in \{1, \ldots, n\}$. We define the system of affine equations $S^i_\alpha$ over $\tau_\dagger \oplus \tau_\dagger$ as follows:

$$\begin{cases} \sigma = \begin{cases} 1 \text{ if } \sigma = \iota^2_1(o_\alpha(\Omega), x, o_\alpha(\Omega')) \ \text{ or } \sigma = \iota^2_1(o_\alpha(\Omega'), f_i, e^i_\alpha(t(\pi_i))) \\ 1 \text{ if } \sigma = \iota_1 \circ \overline{e}(\pi_i) \\ 0 \text{ otherwise} \end{cases} \\ \sigma \in \tau_\dagger \oplus \tau_\dagger \end{cases}$$

We define the element $\rho^\sharp_{\alpha,i}$ of $\mathbf{R}^\sharp(\mathcal{A}_\dagger)$ as follows, where we put $I_i \overset{\text{def}}{=} i(\pi_i)$ and $T_i \overset{\text{def}}{=} t(\pi_i)$:

$$\rho^\sharp_{\alpha,i}(r) \overset{\text{def}}{=} \begin{cases} \mathbf{Sol}^\sharp(S^i_\alpha) \text{ if } \ r = ((\iota^1_0(e(I_i)), \iota^1_0(e(T_i))), (\iota^2_0(o_\alpha(\Omega)), \iota^2_0(e^i_\alpha(T_i)))) \\ \bot^\sharp_{\tau_\dagger \oplus \tau_\dagger} \quad \text{otherwise} \end{cases}$$

Finally we put

$$\{\!|x := f(x_1, \ldots, x_n)|\!\}^\sharp(\mathcal{A}, \rho^\sharp) \overset{\text{def}}{=} (\mathcal{A}_\dagger, [\mathcal{V}^\sharp(\overline{e} \oplus \overline{e})(\rho^\sharp_\bullet) \sqcup^\sharp \bigsqcup_{\alpha \in P_\bullet}{}^\sharp \bigsqcup_{1 \leq i \leq n}{}^\sharp \rho^\sharp_{\alpha,i}]^\sharp_{\mathcal{A}_\dagger})$$

□

**Theorem 18 Soundness of the abstract assignment.** *Let $\alpha$ be an assignment instruction. For any $(\mathcal{A}, \rho^\sharp) \in \mathcal{D}^\sharp$ and any $(L, \equiv) \in \mathrm{g} \circ \gamma(\mathcal{A}, \rho^\sharp)$, we have:*

$$\{\!|\alpha|\!\}(L, \equiv) \in \mathrm{g} \circ \gamma(\{\!|\alpha|\!\}^\sharp(\mathcal{A}, \rho^\sharp))$$

*where $\mathrm{g}$ is the concretization function from $\mathcal{D}$ into $\mathcal{D}^\natural$ defined previously.*

It remains to define a widening on the category $\mathbb{A}$. The idea is to fold states in an automaton that "look similar". This is achieved via the *quotient* of an automaton by an equivalence relation.

**Definition 19 Quotient of an automaton.** Let $\mathcal{A} = (Q, I, \tau)$ be an automaton of $\mathbb{A}$ and $\sim$ be an equivalence relation on $Q$. We denote by $\pi_\sim \in Q \longrightarrow Q/_\sim$ the canonical projection onto the quotient set[2]. We define $\mathcal{A}/_\sim \overset{\text{def}}{=} (Q/_\sim, \pi_\sim(I), \tau_\sim)$ where $\tau_\sim \overset{\text{def}}{=} \{(\pi_\sim(q), a, \pi_\sim(q')) \mid (q, a, q') \in \tau\}$. □

Let $\mathcal{A} = (Q, I, \tau)$ be an automaton and $k \geq 1$ be a fixed integer. For any $q \in Q$ we denote by $\Lambda_q^k(\mathcal{A})$ the set of words $w \in A^*$ of length less than or equal to $k$ such that there exists a path originating from $q$ in the automaton which is labelled by $w$. We define $\equiv_\nabla^k$ to be the least equivalence relation on $Q$ such that:

- $(i \in I) \wedge (i' \in I) \Longrightarrow (i \equiv_\nabla^k i')$.
- $((q, a, q_1) \in \tau) \wedge ((q, a, q_2) \in \tau) \Longrightarrow (q_1 \equiv_\nabla^k q_2)$.
- $(\Lambda_q^k(\mathcal{A}) = \Lambda_{q'}^k(\mathcal{A})) \wedge (\Lambda_q^k(\mathcal{A}) \cap A^k \neq \emptyset) \Longrightarrow (q \equiv_\nabla^k q')$.

Now let $\mathcal{A}_1 = (Q_1, I_1, \tau_1)$, $\mathcal{A}_2 = (Q_2, I_2, \tau_2)$ be two automata, $\mathcal{A}$ their coproduct and $\mathcal{A}_1 \xrightarrow{\epsilon_1} \mathcal{A}$, $\mathcal{A}_2 \xrightarrow{\epsilon_2} \mathcal{A}$ the canonical inclusion maps. We define

$$\mathcal{A}_1 \nabla_k \mathcal{A}_2 \overset{\text{def}}{=} \mathcal{A}/_{\equiv_\nabla^k}$$

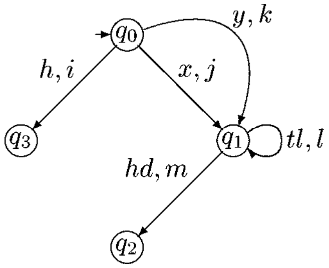
We can extend the projection $\pi_{\equiv_\nabla^k}$ to a morphism of automata $\overline{\pi}_{\equiv_\nabla^k} \in \mathcal{A} \longrightarrow \mathcal{A}_1 \nabla_k \mathcal{A}_2$. For $i \in \{1, 2\}$, we define

$$(\mathcal{A}_1 \nabla_k \mathcal{A}_2)_i \overset{\text{def}}{=} \overline{\pi}_{\equiv_\nabla^k} \circ \epsilon_i$$

In other words, we make $\mathcal{A}$ deterministic and we fold two states that cannot be distinguished by only looking at prefixes of length at most $k$ of the paths originating from them. This widening criterion is inspired from the *k-limiting* approximation of [JM81].

**Proposition 20.** *For any $k \in \mathbb{N}$, $\nabla_k$ is a widening operator on $\mathbb{A}$.*

*Example 7.* Consider the program of Example 5. We use Karr's abstract numerical domain. Since it satisfies the ascending chain condition the widening is given by the join. We use the widening $\nabla_1$ on automata. The analysis computes the element $(\mathcal{A}, \rho^\sharp)$ of $\mathcal{D}^\sharp$. The automaton $\mathcal{A}$ is given by the following diagram[3] where a distinct name in the set $\tau = \{i, j, k, l, m\}$ has been assigned to each transition:



[2] Strictly speaking $Q/_\sim$ is a representative of the quotient set in $Q$.

[3] The auxiliary variables introduced to guarantee the linearity of assignment instructions have been removed. We have also trimmed the automata produced during the analysis by eliminating all useless transitions.

If we denote by $\tau \xrightarrow{\iota_1} \tau \oplus \tau \xleftarrow{\iota_2} \tau$ a coproduct diagram in **Finset**, $\rho^\sharp((q_0, q_2), (q_0, q_2))$ is given by the following system of affine equations:

$$\begin{cases} \iota_1(j) = \iota_2(k) = 1 \\ \iota_2(j) = \iota_1(k) = 0 \\ \iota_1(l) = \iota_2(l) \\ \iota_1(m) = \iota_2(m) = 1 \\ \iota_1(i) = \iota_2(i) = 0 \end{cases}$$

This means that no spurious alias relation has been inferred. □

## 6 Conclusion

We have described the core of an alias analysis for untyped programs based upon cofibered domains. Deutsch's framework [Deu92b, Deu94] can be applied to such programs by extracting datatype declarations from the results of a first analysis phase (using for example a grammar-based analysis [CC95]). However the precision of the alias information heavily relies on the approximation of access paths in data structures by a regular automaton. A separate analysis would produce poor results whenever the alias information interfers with the control flow (pointer equality tests, closures). This is especially obvious for mobile processes where the evolution of a program entirely depends on the sharing of communication ports. The techniques described in this paper have been successfully applied to design an analysis of the communications in the $\pi$-calculus [Ven96]. Work in progress investigates the application of cofibered domains to analyses based upon the class of context-sensitive tree grammars introduced in [CC95].

**Acknowledgements:** I am grateful to Radhia Cousot for helpful comments on first versions of this work. All diagrams in this paper have been designed using Paul Taylor's and Paul Gastin's LaTeX packages.

## References

[Bou92] F. Bourdoncle. Abstract interpretation by dynamic partitioning. *Journal of Functional Programming*, 2(4), 1992.

[BW90] M. Barr and C. Wells. *Category Theory for Computing Science*. Prentice Hall, 1990.

[CC76] P. Cousot and R. Cousot. Static determination of dynamic properties of programs. In *Proceedings of the* $2^{nd}$ *International Symposium on Programming*, pages 106–130, Paris, 1976. Dunod.

[CC77] P. Cousot and R. Cousot. Abstract interpretation : a unified lattice model for static analysis of programs by construction or approximation of fixpoints. In *Conference Record of the* $4^{th}$ *ACM Symposium on Principles of Programming Languages*, pages 238–252, Los Angeles, California, U.S.A., 1977.

[CC79] P. Cousot and R. Cousot. Systematic design of program analysis frameworks. In $6^{th}$ *POPL*. ACM Press, 1979.

[CC92a] P. Cousot and R. Cousot. Abstract interpretation frameworks. *Journal of logic and computation*, 2(4):511–547, August 1992.

[CC92b] P. Cousot and R. Cousot. Comparing the Galois connection and widening/narrowing approaches to abstract interpretation. In M. Bruynooghe and M. Wirsing, editors, *Programming Language Implementation and Logic Programming, Proceedings of the Fourth International Symposium, PLILP'92*, volume 631 of *Lecture Notes in Computer Science*, pages 269–295, Leuven, Belgium, August 1992. Springer-Verlag, Berlin, Germany, 1992.

[CC95] P. Cousot and R. Cousot. Formal language, grammar and set-constraint-based program analysis by abstract interpretation. In *Conference Record of FPCA'95.* ACM Press, 1995.

[CH78] P. Cousot and N. Halbwachs. Automatic discovery of linear restraints among variables of a program. In $5^{th}$ *POPL.* ACM Press, 1978.

[Cou78] P. Cousot. *Méthodes itératives de construction et d'approximation de points fixes d'opérateurs monotones sur un treillis, analyse sémantique des programmes.* PhD thesis, Université Scientifique et Médicale de Grenoble, 1978.

[Cou96] P. Cousot. Abstract interpretation in categorical form. To appear, 1996.

[Deu92a] A. Deutsch. *Operational models of programming languages and representations of relations on regular languages with application to the static determination of dynamic aliasing properties of data.* PhD thesis, University Paris VI (France), 1992.

[Deu92b] A. Deutsch. A storeless model of aliasing and its abstraction using finite representations of right-regular equivalence relations. In *Proceedings of the 1992 International Conference on Computer Languages*, pages 2–13. IEEE Computer Society Press, Los Alamitos, California, U.S.A., 1992.

[Deu94] A. Deutsch. Interprocedural may-alias analysis for pointers : beyond k-limiting. In *ACM SIGPLAN'94 Conference on Programming Language Design and Implementation.* ACM Press, 1994.

[Gra89] P. Granger. Static analysis of arithmetical congruences. *International Journal of Computer Mathematics*, 30:165–190, 1989.

[Gra91] P. Granger. Static analysis of linear congruence equalities among variables of a program. In *TAPSOFT'91*, volume 493. Lecture Notes in Computer Science, 1991.

[Gra92] P. Granger. Improving the results of static analyses of programs by local decreasing iterations. In $12^{th}$ *Foundations of Software Technology and Theoretical Computer Science*, Lecture Notes in Computer Science. Springer Verlag, 1992.

[JM81] N. Jones and S. Muchnick. Flow analysis and optimization of lisp-like structures. In *Program Flow Analysis: Theory and Applications*, pages 102–131. Prentice Hall, 1981.

[Jon81] H.B.M Jonkers. Abstract storage structures. In De Bakker and Van Vliet, editors, *Algorithmic languages*, pages 321–343. IFIP, 1981.

[Kar76] M. Karr. Affine relationships among variables of a program. *Acta Informatica*, pages 133–151, 1976.

[Kel74] G.M. Kelly. On clubs and doctrines. In A. Dold and B. Eckmann, editors, *Category seminar*, volume 420 of *Lecture Notes in Mathematics*, pages 181–256. Springer Verlag, 1974.

[Ven96] A. Venet. Abstract interpretation of the $\pi$-calculus. $5^{\text{th}}$ LOMAPS Workshop on Analysis and Verification of High-Level Concurrent Languages, 1996.